UDC 004.04

RESEARCH AND DEVELOPMENT OF A METHOD FOR EMOTIONAL ANALYSIS OF TEXT DATA

Sarafanov Mykhailo
Senior lecturer, PhD Manikaeva Olga
Odesa Polytechnic National University, UKRAINE

ANNOTATION. This research provides a method of performing sentiment analysis of text information using machine learning. I have applied a supervised learning approach to train a classifier based on the BERT encoder The model recognizes emotions from text messages in English written in chat style. Classifier was trained and tested on the GoEmotions dataset.

Introduction. Supervised learning is widely used for solving various text-classification problems. However, how is it possible to check if the model can understand the text? This can be evaluated using the General Language Understanding Evaluation (GLUE) benchmark [3]. One of the models tested is BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding [1]. The model itself does not solve any problem. However, the pre-trained model can be fine-tuned with additional layers for various language tasks. This study aims to apply the BERT encoder for training a classifier for the emotional analysis of text information. Modeling was performed on the GoEmotions dataset [2].

The purpose of the work. The work aims to create an emotions classifier from text data. The classifier must be able to predict all emotions it was trained on and show better classification quality than presented in the original GoEmotions study. Classification quality is evaluated using the F1-Score metric.

The main part of the work. First of all, before any modeling, it is necessary to research what we are modeling. I have decided to use the GoEmotions dataset [2]. This dataset contains text comments extracted from the Reddit platform. Reddit is a social media platform for discussion, where people share their interests and run discussions on different topics. There are 54,263 comments included in the dataset already divided by the authors into three parts for modeling: training (43,410 elements), validation (5,426), and testing (5,427). All three parts were combined for review.

According to the paper, all comments were manually labeled with 27 emotion categories: admiration, amusement, anger, annoyance, approval, caring, confusion, curiosity, desire, disappointment, disapproval, disgust, embarrassment, excitement, fear, gratitude, grief, joy, love, nervousness, optimism, pride, realization, relief, remorse, sadness, surprise, all of which are divided into three sentiments: positive, negative and ambiguous. Some of the texts are labeled with more than one category.

It is common to express different emotions in a single message, for example, surprise and confusion: «Wow! I didn't expect this to happen. What should we do next?» which means that the second approach would reduce model accuracy and usability. Consequently, the model needs to provide independent predictions of each emotion category and thus should predict more than one emotion for a text piece. The neutral class was removed from the dataset, meaning that the text lacks any emotional characteristics and belongs to the neutral class if no emotions were triggered.

The dataset was re-split into 3 parts: training, validation, and testing. The fraction of the training part was 85% and was applied to each emotion class individually. Additionally, the testing dataset was set up to include only single-labeled elements, so we could create a confusion matrix for all classes using the all-vs-all strategy. The target variable was encoded using the “multi-hot” approach, so the model solves binary classification problem for each emotion category.

The GoEmotions dataset [2] is imbalanced and some of the emotions contain too small a quantity of elements. As a result, during the modeling experiments, the classifier could not predict grief, pride, relief, and embarrassment. I have applied the oversampling technique to resolve this issue by duplicating elements belonging to low classes on the training dataset. The low quantity threshold was set to 500, so random elements were duplicated to reach this quantity.

Training a high-quality model requires operating with clean data. The GoEmotions dataset [2] contains comments written in English in chat style. Considering this, we need to perform text standardization to make the pre-trained model better understand the text. I have implemented a text standardization layer as a part of the classifier so that the complete model can operate with raw unprocessed text.

After a detailed dataset investigation, I have added the following manipulations to the text standardization:

1. Remove non-ASCII characters using the unidecode library [4]. Characters are replaced with their standard form: “ä” becomes “a”
2. Remove placeholders [NAME] and [RELIGION] from the text.
3. Change English contractions [5] with their full spelling where it is unambiguous
4. Letters 'a', 'e', 'i', 'o', 'u', 'y', 's', 'h', 'f', 'r', or 'm' repeated more than three times are replaced with only one. As a result, the message “Loooool I didn’t know that it’s ridiculous” becomes “Lol I didn’t know that it’s ridiculous”.
5. Replace abbreviations and chat words with their full phrases and meaning [6]. For instance, “ASAP” is replaced with “as soon as possible”, while “L8R” becomes “later” and so on.
6. Proper-set punctuation, remove duplications, remove duplicated spaces, and trim text.
7. Convert text to lowercase.

The created text standardization layer is included in the model and can be used for further work in other natural language processing models. I have used the following structure of the classifier:

1. Text standardization layer
2. Preprocess: text preprocess layer added according to the selected BERT model
3. BERT encoder: used small-BERT with 2 layers, 128 hidden units, and 2 attention heads, uncased version
4. Dropout of 40% of neurons
5. Internal dense layer of 256 neurons with ReLU activation function
6. Dropout of 40% of neurons
7. Output dense layer with 27 neurons and sigmoid activation function.

The dataset was batched by 64 elements. The model was compiled with the AdamW optimizer [7], using a schedule for linear decay of a notional initial learning rate (3e-4), prefixed with a linear warm-up phase over the first 10% of training steps. The training was up to 10 epochs, early stop after 2 epochs without validation loss decreased. Figure 1 shows a change in loss and metrics on validation data over each epoch.

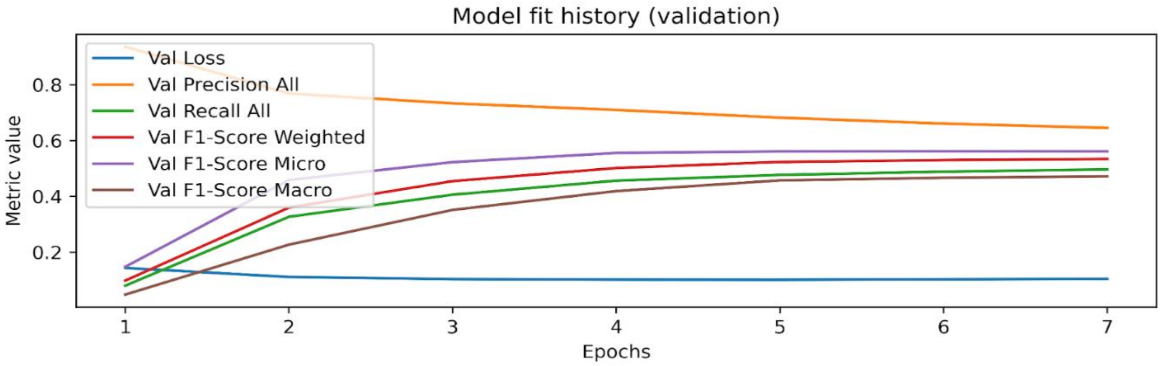

Figure 1 - Change of values of the loss function, Precision, Recall, F1-Score with micro, macro, and weighted averaging for all classes on the validation dataset over train epochs

Finally, the model was evaluated on the testing dataset. Precision and recall for each emotion were non-zero, showing that the classifier can predict and distinguish all emotions it was trained on. F1-Score metrics on the testing dataset for both emotions and sentiment prediction are shown in Table 1. Emotions were converted to sentiments using a map supplied with the GoEmotions dataset [2]. Additionally, I created micro- and macro-averaged ROC curves, which are presented in Figure 2.

In the original GoEmotions study authors also presented results of applying BERT to the dataset [2]. The macro-averaged F1-Score for emotions prediction was 0.46 and for sentiments prediction 0.69.

Table 1- Classification metrics in emotions and sentiments prediction on the testing dataset

	F1-Score micro-averaged	F1-Score macro-averaged
emotions	0.58353	0.50704
sentiments	0.7760	0.7349

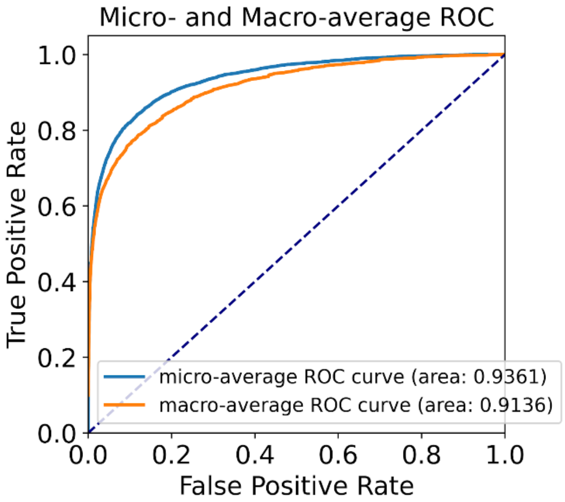

Figure 2 - Micro- and macro-averaged ROC Curve on the testing dataset

Additional analysis shows that the classification quality significantly depends on the amount of data. The model was better at predicting classes having more data for training and testing, such as admiration, amusement, gratitude, and love. At the same time, it made less accurate predictions for embarrassment, relief, excitement, and remorse, which have comparably less amount of labels. Sometimes model confuses semantically close classes. For example, anger and annoyance, curiosity and confusion.

Conclusions. Created emotions classifier based on BERT [1] encoder showed its ability to predict all 27 emotions from the text data. The model has reached F1-Score higher by 8.69% in emotions prediction and 6.5% in sentiments prediction. Experimenting with a more complex BERT encoder and a lower learning rate should increase classification performance. Future work will aim to improve the emotions classification of the text data using a deep learning approach.

REFERENCES

1.arXiv:1810.04805 Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. URL: https://doi.org/10.48550/arXiv.1810.04805 – (date of the application: 10.03.2023).

2.arXiv:2005.00547 Demszky, D., Ko, J., Cowen, A., Nemade, G., & Ravi, S. (2020). GoEmotions: A Dataset of Fine-Grained Emotions. URL: https://doi.org/10.48550/arXiv.2005.00547 – (date of the application: 10.03.2023).

3.arXiv:1804.07461 Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. R. (2018). GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. URL: https://doi.org/10.48550/arXiv.1804.07461 – (date of the application: 10.03.2023).

4.PyPI: Unidecode package ASCII transliterations of Unicode text. URL: https://pypi.org/project/Unidecode/ – (date of the application: 10.03.2023).

5.List of English contractions. URL: https://en.wikipedia.org/wiki/Wikipedia:List_of_English_contractions – (date of the application: 10.03.2023).

6.Kaggle: Getting started with Text Preprocessing, Chat Words Conversion. URL: https://www.kaggle.com/code/sudalairajkumar/getting-started-with-text-preprocessing – (date of the application: 10.03.2023).

7.arXiv:1711.05101 Loshchilov, I., & Hutter, F. (2017). Decoupled Weight Decay Regularization. URL: https://doi.org/10.48550/arXiv.1711.05101 – (date of the application: 10.03.2023).